18482

RECEIVED AT NIC AUG 22, 1973

RECEIVED
SEP 18 1973

Postel

Network Working Group
Request For Comments #559
NIC # 18482

Abhay K. Bhushan
MIT Project MAC
August 15, 1973

Comments on the New TELNET Protocol and its Implementation

We at MIT-DM have implemented the new TELNET protocol (both server and user). This RFC describes our experience with the implementation (particularly the use of GO AHEAD) and in bringing the new User and Server TELNETs in operation (the new TELNET is not compatible with the old). We have a few suggestions here which should help other implementors and lead to a smoother transition to the new protocol.

I. OUR TELNET SERVER IMPLEMENTATION

Our new server TELNET accepts both the "old" and the "new" TELNET "control sequences". Currently we have the ECHO and the 'Suppress Go Ahead" options implemented and do the "right thing" to varying degrees with the Interrupt Process (IP), Erase Character (EC), Abort Output (AO), Are You There (AYT), Break, and Synch character sequences.

A. The ECHO Option

The TELNET server comes up in the default local echo mode and accepts both the old and the new TELNET control sequences. The server starts the negotiation for remote echo (server echoing) by sending the sequence "IAC WILL ECHO" but changes the echo mode only when an affirmative "IAC DO ECHO" is received. After the cutoff date for old protocol we will stop "honoring" the old TELNET control sequences.

B. The Go Ahead and Suppress GA option

The server comes up in the send GA mode and transmits the required "IAC GA" sequence whenever the server detects that it needs to send a GA. It should be noted that our scheme for sending GA's works most but not all of the time.

There is really no reliable way for our server TELNET to find out when a process is actually waiting for network input. On our system, the server TELNET does not "control" user's process, it only acts as a terminal handler at the TTY level.

A quick investigation revealed that the above problem (of sending GA's reliably) is not confined to the ITS operating system alone. In fact TENEX (ref. conversation with Ray Tomlinson) and DEC-10 (ref. conversation with Ed Taft at Harvard) systems will encounter similar problems.

Our solution to the GA sending problem wasto have the server wait 2.5 seconds after sending output to see if there is more output to be sent. If the server has been "idle" for more than 2.5 seconds in the "output-sent" state it sends a GA and goes in an I/O wait state (looking for input or output). This scheme works most (but not guaranteed all) of the time and doesn't cause any noticable delay. It is possible for the server to send an extra GA. Our experimentation revealed that 1-5 seconds was a good range for this "idling time constant".

We do implement the "suppress GA " option and will not send GA to hosts who agree to negotiate out of it. Our server tries to negotiate the suppress GA option.

C. Other Options and TELNET Control Sequences

Our server will refuse all other options by sending the appropriate DONTs and WONTs. In addition to the ECHO and Suppress GA options we recognize the following TELNET "control sequences".

1. Interrupt Process (IP) - The server substit utes the system wide interrupt character <control-Z> (ASCII SUB) which immediately interrupts the process, moving control to the immediately superior process. If the user is several levels down his process tree he may have to send several IP's to reach top level. It should be noted that the IP does not interrupt the running process in the sense a <control-G> interrupts muddle but only passes control to the superior.

2. Erase Character (EC) - The server substitutes the system wide standard erase character <rubout> (ASCII DEL>. The deletion however is done not by the server but by the receiving process. It is conceivable that some processes (such as a user TELNET) take no action on receiving EC. Most processes will echo the deleted character(s). Several EC's will delete the several previous characters. (If the console is declared to be an IMLAC, the deleted character is removed from the screen).

3. Abort Output (AO) - The server substitutes the character <control-S> (ASCII DC3). The control-S convention is followed by many but not all of our programs. The action taken on receiving AO varies with the program. A normal occurence is that output and the current command are aborted (without necessarily going to completion). In many programs there is no way to stop output except by sending an IP and "killing" the inferior process.

4. Are You There (AYT) - The server will print the message "****connections still open*****" preceded and followed by CRLF's upon receing an AYT. At some later time we may report on the state of the user's job as well.

5. Erase Line (EL) - since we are a character-at-a-time system, the EL has little meaning on our system and we throw it (and the preceding IAC) away.

6. Break (BRK) - We substitute three NUL's upon receiving BRK. This convention is consistent with what happens when the "Break" key is hit on local teletypes. The programs generally do nothing useful when break is received (except echo "↑@↑@↑@") but sending BRK may cause strange program reactions, so beware.

7. Synch - Whenever the server receives the synch IMS, it flushes all except the interesting (control sequences) characters till the receipt of a DM. We also cause an implicit IP on receipt of SYNCH.

8. We follow the CRLF and CRNUL convention for transmitting EOL and CR respectively.

II. OUR USER-TELNET IMPLEMENTATION

The new user-TELNET (implemented in CALICO NETWRK by using a new "TELCOM" subroutine), accepts negotiation for the ECHO and suppress GA options. The program tries to negotiate out of receiving GA's and tries the ECHO negotiation if the settings file for the host indicate remote echo. Special characters and symbols are defined for EC, EL, AO, AYT, BRK, SYNCH, IP, and the ESCAPE character to command level. These symbols have a default character value which the user can change by typing the symbol followed by the new character value at NETWRK command level. To send EC, EL, etc, the user only has to type the special character for the function. In addition the user can send these characters by using the send.special command at NETWRK command llevel. In "line mode", EC and EL do a "local" character and line erase rather than send the EC and EL to the remote Host. The following are the default values for the "special" characters in TELNET:

ESCAPE - backslash
EC - <DEL>
EL - <CAN> or ↑X
AO - ↑S
IP - ↑R
AYT - ↑T
Synch - ↑Y
Break - no preassigned value.

The user can change his echo mode by escaping to NETWRK command level and using the commands "echo.local" or "echo.remote". Note that the modes are changed only when the negotiation for mode change is succcessful. In either event the user is notified of the results of the negotiation.

III. INSTALLING THE NEW TELNETS

On Monday July 2, we brought up the user and server TELNETs briefly to find that most of the hosts did not "recognize" IAC's and did not honor the new protocol. Much to our dismay usage of both our server and user TELNETs was chaotic. Consequently, we did not install the new user and server TELNETs, and the old TELNETs remain operational.

The new and the old TELNETs are definitely not compatible. The server tries to (and should try to) negotiate out of sending GA's and also to send echo. This negotiation causes problems with the "old-style" user TELNETs. Also if the negotiation for Suppress GA is unsuccessful (which is the case with "old" user-TELNETs) the server will keep sending IAC GA's when appropriate. One solution we found to making our "new" server compatible with "old" user TELNETs was to come up in a mode that does not start any option negotiation and does not send GA's unless requested to do so (ie default is to suppress GA's). As mentioned earlier the server also accepts the old "TELNET control" sequences. This solution makes the server compatible with both the old and the new user TELNETs (except it violates protocol by not sending GA's). We propose to install this server on our socket 1.

To promote experimentation with the new TELNET protocol, we have installed the new server TELNET on socket 69 (octal 105). This new server follows the new protocol (ie it sends GA's) and starts spontaneous negotiation for remote echo and suppress GA. The implementors on other Hosts are encouraged to use this service to debug their user TELNETs (and our server). We feel that transition to the new protocol will be smoother if new TELNET servers are brought up on experimental sockets. We are also willing to help debug other servers from our User TELNET.

Finally we would like to lobby for making suppress GA the default (as our present server on socket 1). It appears that only a few Hosts require the GA's (AMES-67 and UCLA-CCN). It seems to me that the reason why sending GA is default in the current specification of protocol is that representatives from the concerned Hosts wanted the GA to be implemented. It doesn't matter to them if sending GA or suppress GA is default, as long as they can get a remote server to send a GA. The protocol can be so specified as to require every one to implement a "send GA option". Making "suppress GA" the default will have the

advantage that it will obviate unnecessary negotiation in a great majority of cases. Another advantage is that not sending GA's makes the new server compatible with both old and new user TELNETs.